Shuffling

Nipun Batra

IIT Gandhinagar

July 31, 2025

Outline

Why Shuffle? A Motivating Example

The Problem

Question: What happens if our data has hidden patterns or ordering?

Why Shuffle? A Motivating Example

The Problem

Question: What happens if our data has hidden patterns or ordering?

Why Shuffle? A Motivating Example

The Problem

Question: What happens if our data has hidden patterns or ordering?

Let's see with a concrete example...

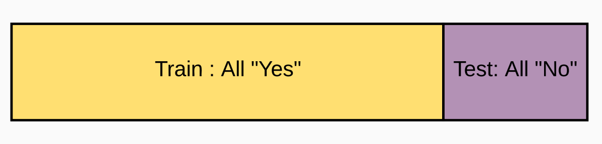
Consider this dataset

First 80 examples are of class "Yes"
Remaining 20 examples are of class "No".

Serial Number	**...**	**Class**
1		Yes
2		Yes
3		Yes
. .		. .
80		Yes
81		No
. .		. .
100		No

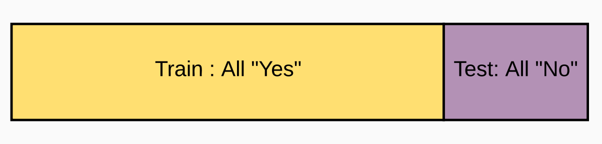
The Disaster: Ordered Data Split

With 80-20 train-test split on ordered data:



The Disaster: Ordered Data Split

With 80-20 train-test split on ordered data:

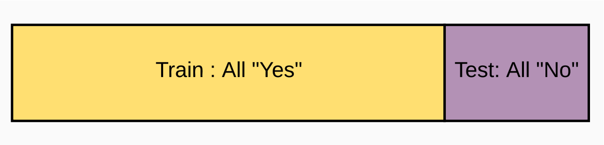


Critical Question

Will we learn anything useful in this scenario?

The Disaster: Ordered Data Split

With 80-20 train-test split on ordered data:



Critical Question

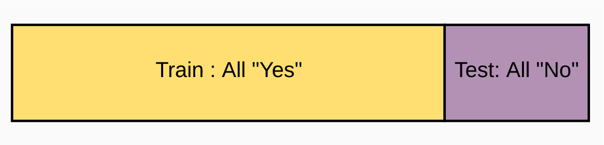
Will we learn anything useful in this scenario?

Answer: NO!

- Training set: Only "Yes" examples
- Test set: Only "No" examples

The Disaster: Ordered Data Split

With 80-20 train-test split on ordered data:



Critical Question

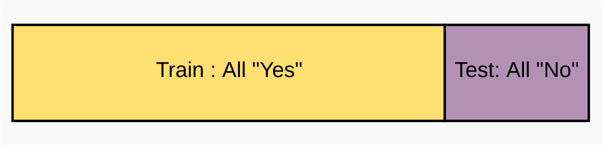
Will we learn anything useful in this scenario?

Answer: NO!

- Training set: Only "Yes" examples
- Test set: Only "No" examples
- Model learns: "Always predict Yes"

The Disaster: Ordered Data Split

With 80-20 train-test split on ordered data:



Critical Question

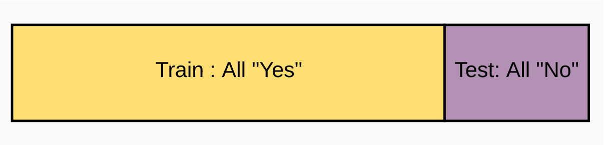
Will we learn anything useful in this scenario?

Answer: NO!

- Training set: Only "Yes" examples
- Test set: Only "No" examples
- Model learns: "Always predict Yes"
- Test accuracy: 0%!

The Disaster: Ordered Data Split

With 80-20 train-test split on ordered data:



Critical Question

Will we learn anything useful in this scenario?

Answer: NO!

- Training set: Only "Yes" examples
- Test set: Only "No" examples
- Model learns: "Always predict Yes"
- Test accuracy: 0%!

Pop Quiz: Data Ordering

Quick Quiz 1

You have a time-series dataset with data points ordered by time. For machine learning, you should:

a) Keep the time order to preserve patterns

Answer: c) For time-series prediction, order matters! For general classification, shuffle.

Pop Quiz: Data Ordering

Quick Quiz 1

You have a time-series dataset with data points ordered by time. For machine learning, you should:

a) Keep the time order to preserve patterns
b) Always shuffle to avoid bias

Answer: c) For time-series prediction, order matters! For general classification, shuffle.

Pop Quiz: Data Ordering

Quick Quiz 1

You have a time-series dataset with data points ordered by time. For machine learning, you should:

a) Keep the time order to preserve patterns
b) Always shuffle to avoid bias
c) It depends on the problem

Answer: c) For time-series prediction, order matters! For general classification, shuffle.

SGD Without Shuffling: The Oscillation Problem

The Problem

Without shuffling, SGD can get stuck in cycles!

SGD Without Shuffling: The Oscillation Problem

The Problem

Without shuffling, SGD can get stuck in cycles!

Example with 2 data points:

- **Step 1 - Point 1:** $\theta_0 \leftarrow \theta_0 + 0.2, \theta_1 \leftarrow \theta_1 - 0.2$

SGD Without Shuffling: The Oscillation Problem

The Problem

Without shuffling, SGD can get stuck in cycles!

Example with 2 data points:

- **Step 1 - Point 1:** $\theta_0 \leftarrow \theta_0 + 0.2, \theta_1 \leftarrow \theta_1 - 0.2$

SGD Without Shuffling: The Oscillation Problem

The Problem

Without shuffling, SGD can get stuck in cycles!

Example with 2 data points:

- **Step 1 - Point 1:** $\theta_0 \leftarrow \theta_0 + 0.2, \theta_1 \leftarrow \theta_1 - 0.2$
- **Step 2 - Point 2:** $\theta_0 \leftarrow \theta_0 - 0.2, \theta_1 \leftarrow \theta_1 + 0.2$

SGD Without Shuffling: The Oscillation Problem

The Problem

Without shuffling, SGD can get stuck in cycles!

Example with 2 data points:

- **Step 1 - Point 1:** $\theta_0 \leftarrow \theta_0 + 0.2, \theta_1 \leftarrow \theta_1 - 0.2$
- **Step 2 - Point 2:** $\theta_0 \leftarrow \theta_0 - 0.2, \theta_1 \leftarrow \theta_1 + 0.2$

SGD Without Shuffling: The Oscillation Problem

The Problem

Without shuffling, SGD can get stuck in cycles!

Example with 2 data points:

- **Step 1 - Point 1:** $\theta_0 \leftarrow \theta_0 + 0.2, \theta_1 \leftarrow \theta_1 - 0.2$
- **Step 2 - Point 2:** $\theta_0 \leftarrow \theta_0 - 0.2, \theta_1 \leftarrow \theta_1 + 0.2$
- **Step 3 - Point 1:** Back to step 1...

SGD Without Shuffling: The Oscillation Problem

The Problem

Without shuffling, SGD can get stuck in cycles!

Example with 2 data points:

- **Step 1 - Point 1:** $\theta_0 \leftarrow \theta_0 + 0.2, \theta_1 \leftarrow \theta_1 - 0.2$
- **Step 2 - Point 2:** $\theta_0 \leftarrow \theta_0 - 0.2, \theta_1 \leftarrow \theta_1 + 0.2$
- **Step 3 - Point 1:** Back to step 1...

Why This Happens

Biased learning: Point 2 always follows Point 1, creating predictable oscillations

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

Benefits of shuffling:

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

Benefits of shuffling:

- **Breaks cycles:** No predictable patterns in gradient updates

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

Benefits of shuffling:

- **Breaks cycles:** No predictable patterns in gradient updates

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

Benefits of shuffling:

- **Breaks cycles:** No predictable patterns in gradient updates
- **Better convergence:** More diverse gradient directions

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

Benefits of shuffling:

- **Breaks cycles:** No predictable patterns in gradient updates
- **Better convergence:** More diverse gradient directions

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

Benefits of shuffling:

- **Breaks cycles:** No predictable patterns in gradient updates
- **Better convergence:** More diverse gradient directions
- **Reduces bias:** Each sample gets fair treatment

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

Benefits of shuffling:

- **Breaks cycles:** No predictable patterns in gradient updates
- **Better convergence:** More diverse gradient directions
- **Reduces bias:** Each sample gets fair treatment

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

Benefits of shuffling:

- **Breaks cycles:** No predictable patterns in gradient updates
- **Better convergence:** More diverse gradient directions
- **Reduces bias:** Each sample gets fair treatment
- **Improves generalization:** Model sees varied combinations

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

Benefits of shuffling:

- **Breaks cycles:** No predictable patterns in gradient updates
- **Better convergence:** More diverse gradient directions
- **Reduces bias:** Each sample gets fair treatment
- **Improves generalization:** Model sees varied combinations

The Solution: Random Shuffling

Best Practice for SGD

Shuffle the dataset before each epoch!

Benefits of shuffling:

- **Breaks cycles:** No predictable patterns in gradient updates
- **Better convergence:** More diverse gradient directions
- **Reduces bias:** Each sample gets fair treatment
- **Improves generalization:** Model sees varied combinations

Implementation

`np.random.shuffle(training_data)` before each epoch

Pop Quiz: SGD Shuffling

Quick Quiz 2

How often should you shuffle your training data in SGD?

a) Once at the beginning of training

Answer: b) Before each epoch ensures maximum randomness while being computationally efficient!

Pop Quiz: SGD Shuffling

Quick Quiz 2

How often should you shuffle your training data in SGD?

a) Once at the beginning of training
b) Before each epoch

Answer: b) Before each epoch ensures maximum randomness while being computationally efficient!

Pop Quiz: SGD Shuffling

Quick Quiz 2

How often should you shuffle your training data in SGD?

a) Once at the beginning of training
b) Before each epoch
c) After every batch

Answer: b) Before each epoch ensures maximum randomness while being computationally efficient!

Key Takeaways

- **Data ordering matters:** Hidden patterns can destroy learning

Key Takeaways

- **Data ordering matters:** Hidden patterns can destroy learning

Key Takeaways

- **Data ordering matters:** Hidden patterns can destroy learning
- **Always shuffle:** Before train-test split and during SGD

Key Takeaways

- **Data ordering matters:** Hidden patterns can destroy learning
- **Always shuffle:** Before train-test split and during SGD

Key Takeaways

- **Data ordering matters:** Hidden patterns can destroy learning
- **Always shuffle:** Before train-test split and during SGD
- **Prevents cycles:** SGD oscillations and biased updates

Key Takeaways

- **Data ordering matters:** Hidden patterns can destroy learning
- **Always shuffle:** Before train-test split and during SGD
- **Prevents cycles:** SGD oscillations and biased updates

Key Takeaways

- **Data ordering matters:** Hidden patterns can destroy learning
- **Always shuffle:** Before train-test split and during SGD
- **Prevents cycles:** SGD oscillations and biased updates
- **Simple but critical:** One line of code, huge impact on performance

Key Takeaways

- **Data ordering matters:** Hidden patterns can destroy learning
- **Always shuffle:** Before train-test split and during SGD
- **Prevents cycles:** SGD oscillations and biased updates
- **Simple but critical:** One line of code, huge impact on performance

Key Takeaways

- **Data ordering matters:** Hidden patterns can destroy learning
- **Always shuffle:** Before train-test split and during SGD
- **Prevents cycles:** SGD oscillations and biased updates
- **Simple but critical:** One line of code, huge impact on performance
- **Exception:** Time-series and sequential data require special handling

Key Takeaways

- **Data ordering matters:** Hidden patterns can destroy learning
- **Always shuffle:** Before train-test split and during SGD
- **Prevents cycles:** SGD oscillations and biased updates
- **Simple but critical:** One line of code, huge impact on performance
- **Exception:** Time-series and sequential data require special handling

Key Takeaways

- **Data ordering matters:** Hidden patterns can destroy learning
- **Always shuffle:** Before train-test split and during SGD
- **Prevents cycles:** SGD oscillations and biased updates
- **Simple but critical:** One line of code, huge impact on performance
- **Exception:** Time-series and sequential data require special handling

Remember

Good shuffling is the foundation of good machine learning!